PATENTS
15311-2207
200308268-1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re The Application of: Stephen Dickson	
Serial No.: 09/428,384	Examiner: Ly, Anh
Filed: October 28, 1999	Art Unit: 2162
For: COMPUTERIZED FILE SYSTEM AND METHOD	Confirmation No.: 4583

Cesari and McKenna, LLP
88 Black Falcon Avenue
Boston, MA 02210
November 3, 2008

CERTIFICATE OF EFS WEB TRANSMISSION

I hereby certify that the following paper is being EFS WEB transmitted to the Patent and Trademark Office on November 3, 2008.

/Merisa Jakupovic/
Merisa Jakupovic

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

APPEAL BRIEF

In response to the Notice of Appeal transmitted September 3, 2008, Applicant hereby submits this Appeal Brief.

REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, L.P. of Houston, Texas.

RELATED APPEALS AND INTERFERENCES

Applicant and its legal representatives know of no related appeals or interferences that will directly affect, be directly affected by, or have a bearing on the Board's decision in the present appeal.

STATUS OF CLAIMS

Claims 1-40 are pending in the case. Claims 1-40 stand rejected under 35 U.S.C. §103, as noted in the final Office Action mailed June 13, 2008. The rejection of claims 1-40 is appealed.

A copy of claims 1-40 is attached hereto as an Appendix.

STATUS OF AMENDMENTS

No amendments have been filed since the mailing of the final Office Action on June 13, 2008.

SUMMARY OF CLAIMED SUBJECT MATTER

This summary is set forth in eleven exemplary embodiments that correspond to independent claims 1, 6, 11, 14, 19, 20, 21, 27, 30, 33 and 36. Discussions about elements and recitations of these claims can be found at least at the cited locations in the specification and drawings.

Independent claim 1 is directed to a computerized file system in which a first process (252), which may reside in a server node (202), maintains a data file (250). A second process (260), which may reside at a client node (204), generates a first message (300) that requests that the second process be granted a plurality of tokens that are required to modify at least one characteristic of the data file. The first process (252) generates a second message (302) that grants the tokens to the second process, assuming they are available. The second process modifies the at least one characteristic of the data file, if the tokens are granted, without receiving a copy of the data file. Specification, p. 6, lines 3-9, p. 13, line 1 to p. 15, line 4, and Figs. 3-5.

Independent claim 6 is directed to a computer node (202) including a first process (252) residing in the node that generates a first message (302) that grants a set of tokens to a second process (260) that requested the grant of the set of tokens, where the set of tokens is required for the second process to modify at one characteristic of a file (250). The second process modifies the at least one characteristic of the file without receiving a copy of it. Specification, p. 6, lines 3-9, p. 11, line 12 to p. 15, line 4, and Figs. 2-3.

Independent claim 11 is directed to a computer node (204) including a first process (260) that generates a request to a second process (252) for a set of tokens required to modify at least one characteristic of a file (250). If it receives the tokens, the first process modifies the at least one characteristic without receiving a copy of the file. Specification, p. 6, lines 3-9, p. 13, line 1 to p. 15, line 4, and Figs. 3-5.

Independent claim 14 is directed to a network computing system comprising a first node (202) having a data file (250), and a second computer node (204) that issues a first message (300) requesting the grant of a set of tokens for modifying at least one characteristic of the data file, where the first computer node issues a second message (302) to the second computer node that grants the set of tokens if they are available. The second computer node modifies the at least one characteristic of the file without receiving a copy of it.

Specification, p. 6, lines 3-9, p. 11, line 12 to p. 12, line 11, p. 13, line 1 to p. 15, line 4, and Figs. 2-5.

Independent claim 19 is directed to a computer-readable memory (210) containing computer-executable program instructions including a first instruction that maintains a data file (250) in computer storage memory, a second instruction that generates a first message (300) requesting the grant of a plurality of tokens that are required to modify at least one characteristic of the data file, a third instruction that generates a second message (302) that grants the tokens, assuming they are available, and a fourth instruction modifying the at least one characteristic without the process receiving a copy of the file. Specification, p. 6, lines 3-9, p. 11, line 12 to p. 15, line 4, and Figs. 2-5.

Independent claim 20 is directed to a computer-readable memory (210) containing computer-executable program instructions including first instructions that generate a first message (302) granting a set of tokens to a requester (260) of the set of tokens, where the set of tokens is required to modify at least one characteristic of a file (250), and second instructions that modify the at least one characteristic of the file without the requester receiving a copy of the file. Specification, p. 6, lines 3-9, p. 11, line 12 to p. 15, line 4, and Figs. 2-5.

Independent claim 21 is directed to a computer-readable memory (210) containing computer-executable program instructions including first instructions that generate a request (300) for a grant of a set of tokens required to modify by an issuer (260) of the request at least one characteristic of a file (250), and second instructions that modify the at least one characteristic of the file without the issuer receiving a copy of the file. Specification, p. 6, lines 3-9, p. 11, line 12 to p. 15, line 4, and Figs. 2-5.

Independent claim 27 is directed to a computerized file system including means for maintaining a data file (processor 212, memory 210, and data file 250 of node 202), means for generating a first message (processor 212 and memory 210 of node 204, and message 300) requesting a grant of a plurality of tokens for modifying at least one characteristic of the file, means for generating a second message (processor 212 and memory 210 of node 202, and message 302) in response to the first, that grants the tokens if they are available, and means for modifying the data file (processor 212 and memory 210) if the tokens are granted. Specification, p. 6, lines 3-9, p. 11, line 14 to p. 15, line 4, and Figs. 2-5.

Independent claim 30 is directed to a computerized method including maintaining a data file (250), generating a first message (300) requesting a grant of a plurality of tokens to modify at least one characteristic of the file, generating a second message (302) in response to the first, that grants the tokens if they are available, and modifying the at least one characteristic of the file if the tokens are granted. Specification, p. 6, lines 3-9, p. 11, line 14 to p. 15, line 4, and Figs. 2-5.

Independent claim 33 is directed to a computerized method including generating a first message (302) that grants a set of tokens that are required to modify at least one characteristic of a file (250), and modifying the at least one characteristic of the file if the tokens are granted without the requester receiving a copy of the file. Specification, p. 6, lines 3-9, p. 11, line 14 to p. 15, line 4, and Figs. 2-5.

Independent claim 36 is directed to a computerized method including generating a request (300) for a grant of a set of tokens required to modify at least one characteristic of a file (250), and modifying the at least one characteristic of the file if the tokens are granted without the requester receiving a copy of the file. Specification, p. 6, lines 3-9, p. 11, line 14 to p. 15, line 4, and Figs. 2-5.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1, 6, 11, 14, 19, 20, 21, 27, 30, 33 and 36 are unpatentable under 35 U.S.C. §103 on the grounds that they are obvious over U.S. Patent No. 6,385,658 to Harter, Jr. et al. ("Harter") in view of U.S. Patent No. 5,255,477 to Strickland et al. ("Strickland").

Whether claims 2-5, 7-10, 12, 13, 15-18, 22-26, 28, 29, 31, 32, 34, 35 and 37-40 are unpatentable under 35 U.S.C. §103 on the grounds that they are obvious over Harter in view of Strickland and U.S. Patent No. 5,634,122 to Loucks et al. ("Loucks").

ARGUMENT

Legal Standard

Section 103(a) states:

> A patent may not be obtained … if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.

In rejecting claims under 35 U.S.C. §103, the examiner bears the initial burden of presenting a prima facie case of obviousness. *In re Rijckaert*, 9 F.3d 1531, 1532 (Fed. Cir. 1993). As the Supreme Court recently held, the question of obviousness is resolved on the basis of underlying factual determinations including (1) the scope and content of the prior art, (2) the differences between the claimed subject matter and the prior art, and (3) the level of skill in the art. *KSR Internat'l Co. v. Teleflex Inc.*, 127 S.Ct. 1727, 1734 (2007).

Obviousness cannot be proven merely by showing that the elements of a claimed device were known in the prior art. *Id.* at 1741. Instead, it must be shown that one of ordinary skill in the art would have had some "apparent reason to combine known elements in the fashion claimed." *Id.* Likewise, it can be "important to identify a reason that would have prompted a person of ordinary skill in the art in the relevant field to combine the elements in the way the claimed new invention does." *Id.* at 1741. The *KSR* Court did not discard the teaching-suggestion-motivation (TSM) test, but ruled that a rigid and mandatory application of that test is incompatible with the Court's precedents. *Id.*

The claims do not stand or fall together. Instead Applicant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.F. §41.37(c)(1)(vii).

Scope and Content of the Prior Art

Harter describes a method for communicating messages among processes using a shared message buffer pool. *See* Abstract. To send a message, a message buffer is allocated to the sending process from a free message buffer. *See* Col. 5; 42-51. To receive a message, a receiving process accesses the buffer and processes the received message. The receiving process then returns the buffer including the message that it just processed to the free message buffer list. *See* Col. 5; 65 to Col. 6; 5. In sum, Harter describes a technique for allocating, in an efficient manner, shared memory of a computer system that is used to exchange information among processes that have access to the shared memory. In contrast to the prior techniques in which each process is statically allocated its own message buffer in which to store information to be exchanged with other processes, Harter's improvement involves adaptively allocating a free list in memory as needed, and eliminating any requirement for a global locking mechanism. *See* Col. 3; 17-29.

Strickland describes a system for performing concurrent record updating in a relational database management system. *See* Col. 2; 65 to Col. 3; 7. In particular, Strickland discloses a system for permitting concurrent accesses to a single block of memory by two or more updating processes with serialization by record-level locking alone. *See* Col. 3; 32-33. There are three aspects to Strickland's invention: multiple Private Buffer (PB) data Control Interval (CI) copies, a Shared Local Cache (SLC) invalidation protocol, and a conditional write function. *See* Col. 3; 4-7.

Loucks describes a technique for controlling access to shared resources in a distributed computer system using a token manager. *See* Abstract. In order to access the shared resource, a process (e.g., a client process) must hold a single token associated with the operation. The token represents an authorization for the client process to perform the operation. *See* Abstract, and Col. 6, lines 8-18. If the client process does not hold the token, it requests the token from a token manager. *See* Col. 6, lines 12-16. The token manager examines a list of tokens it holds and, depending on the content of the list, may request a new token from a local token manager. Alternatively, it may request that another client process revoke a token, or it may simply grant an available token to the requesting client process. See Col. 6, lines 28-36.

In summary, with Loucks a process requests a single token from a token manager in order to access a shared resource. The token manager, in turn, either grants the token immediately, requests the token from a local token manager, or revokes the token from another process. Moreover if the requested token is not available, depending on the type of the token being requested, the token manager may, in turn, revoke several other types of tokens from other processes before granting the requested single token.

Level of Skill

Applicant submits that the level of skill in the art is a person having a B.S. degree in computer science or computer engineering and several years work experience in systems providing multiple accesses to data.

Differences Between the Claimed Invention and the Prior Art

Claims 1, 14, 19, 27 and 30

Claim 1 recites as follows:

> 1. A computerized data file system, comprising:
>
> a first process that maintains a data file stored in a computer-readable memory; and
>
> a second process that generates a first message requesting that said second process be granted by said first process a plurality of tokens required for said second process to modify at least one characteristic of said file stored in said computer-readable memory,
>
> > said first process generating a second message, in response to said first message, that grants said tokens to said second process if said tokens are available for grant to said second process, and
> >
> > if said tokens are granted, said second process modifying the at least one characteristic of said data file as maintained by said first process in said computer-readable memory without said second process receiving a copy of said data file.

Single Message Requesting/Granting Multiple Tokens

As shown, claim 1 recites, among other things, that there are a **plurality** of tokens being requested by the second process in the **single first message**, and that there are a plurality of tokens being granted by the first process in the single second message. The final Office Action relies on Harter, the primary reference, as purportedly disclosing or suggesting these limitations. As set forth herein, Applicant respectfully disagrees that Harter or the other prior art of record either alone or in combination renders Applicant's claimed single message that either requests or grants a plurality of tokens required to modify a file obvious.

As mentioned above, in rejecting claim 1, the final Office Action contends that Harter discloses a single message granting a plurality of requested tokens if the tokens are available for grant. In particular, the final Office Action cites to Col. 6; 6-24 of Harter, which states as follows:

> Referring now to FIG. 3, shown is a block diagram of an embodiment of data structures included in FIG. 2. It should be noted that these data structures represent a "snapshot" of the data structures during the execution of process A and process B. Process A's MMI 40 includes a local head pointer 40a, a remote tail pointer 40b and a free list pointer 40c. The local head pointer 40a points to the head, or first message, of the message list for process A 38a. Remote tail pointer 40b identifies the tail end or the last element on process B's message list 38b. The free list pointer 40c identifies the first message buffer in the list of free message buffers 38c available for use by process A when sending a message to process B. Similarly, process B's MMI 42 contains a local head pointer 42a identifying the first message on process B's list of incoming messages 38b. The remote tail pointer 42b identifies the last element on process A's message list 38a. Free list pointer 42c identifies the first message buffer in the list of free pointers 38c available for use by process B when sending a message to process A.

Applicant submits that this excerpt teaches how a section of shared memory may be organized in order to exchange information between two processes that have access to the shared memory. The fields of the data structures illustrated in FIG. 3 of Harter include work link (44c) and state (44d) fields that are used to manage process A's message list (38a), *see* Col. 6; 61-62, and AB_link (44a) and BA_link (44 b) fields that identify message buffers and free lists. *See* Col. 10; 36-45.

The cited excerpt provides no disclosure or even suggestion of a single message granting a plurality of tokens if the tokens are available for grant. Indeed, the excerpt makes no mention of tokens.

The other parts of Harter cited by the final Office Action likewise fail to disclose or even suggest a single message granting a plurality of tokens in response to a request for the plurality of tokens. For example, Col. 6; 45-60 states that:

> Process A's message list 38a shown in FIG. 3 includes two message buffers 44 and 46. Note that, as previously described, this is a "snapshot" of the data structures, such as message list 38a. Taking a "snapshot" of the data structures at other points of execution may result in different data structure states. The first message buffer 44 included in process A's message list 38a is identified by the local head pointer 40a of process A's MMI 40. Message buffer 44 contains an AB link field 44a, a BA link field 44b, a work link field 44c and a state field 44d followed by any additional data for the message to be sent to process A. It should be noted that all message buffers appearing in any of the data structures--process A message list, process B message list or the free message buffer list 38c--include these same fields. However, different fields are used in conjunction with different data structures as will be described in paragraphs that follow.

This excerpt simply describes some of the fields of the data structure, none of which are described as a token, as recited in claim 1.

At Col. 9; 60 to Col. 10; 15, Harter states that:

> Referring now to FIG. 7B, a flowchart of an embodiment of a method for retrieving a message received by process A from process B is shown in which the method uses the data structures previously described in conjunction with FIG. 3. At step 84a, a determination is made using the data structures of FIG. 3 as to whether or not there is more than one message buffer in process A's message list 38a. This is determined by examining the work link field of the message buffer identified by the local head field 40a of process A's MMI data structure 40. If this work link field is NULL, control proceeds to step 85. If this work link field is not NULL, control proceeds to step 86a where a temp pointer is used to identify the first message buffer on process A's message list 38a. At step 86b, the first message buffer in process A's message list is disconnected from the remainder of A's message list by updating A's MMI field local head 40a to point to the next consecutive message buffer in process A's message list. At step 88a, a determination is made as to whether or not the disconnected first message buffer identified by the temp pointer has been processed. This is done by examining the state field of the message buffer identified by the temp pointer.

This excerpt from Harter simply describes how a process retrieves a message, and what is done with the free message list.

Applicant submits that none of the excerpts from Harter discloses or even suggest a single message granting a plurality of requested tokens if the tokens are available for grant. Applicant submits that one skilled in the art, upon reading Harter, would not somehow substitute Harter's message list management fields with tokens required to modify at least one characteristics of a file. The final Office Action, moreover, provides no explanation as to why or how one skilled in the art would make such a leap from reading Harter. Indeed, the term "token" nowhere appears in Harter.

Applicant submits, moreover, that one skilled in the art would not have any reason, upon reading, Harter to come up with the idea of transmitting multiple tokens in a single message. The data contained in Harter's data structure, namely message list management fields, has nothing to do with tokens required to modify at least one characteristic of a file. Furthermore, Harter's data structure stores temporary information, which is deleted or lost when the buffer is released and returned to the free buffer list, whereas the owner of tokens required to modify a characteristic of a file must be maintained, e.g., so that they may be revoked later on. Accordingly, Applicant submits that the rejection of claim 1 should be withdrawn.

Modifying the File Without Receiving a Copy

As set forth above, claim 1 further recites that the second process, if the requested tokens are granted, modifies the at least one characteristic of the file without receiving a copy of the file. The final Office Action cites to Strickland as disclosing this limitation of claim 1. Applicant respectfully disagrees.

In particular, the final Office Action cites to the Abstract, lines 11-16 of Strickland, which states:

> A first process updates a private copy of a data [Control Interval (CI)] containing the target record from [Shared External Storage (SES)], logs the changes, and writes the changed data CI back to the SES conditioned upon no update having been made to the same block data CI by a second process in the interim.

This excerpt discloses that a process updates its own private copy of data, and then writes the changed data back to shared storage. In contrast, claim 1 recites that the second process modifies the at least one characteristic of the data file "without said second process receiving a copy of said data file." (emphasis added)

Applicant submits that the other cited excerpts of Strickland also fail to disclose or even suggest the claimed limitation. For example, Col. 1; 45-54 states:

> In multiprocessing systems generally, and database management systems in particular, when one user accesses a file for editing or updating purposes, all other users are locked out until the accessing update is completed or committed. To improve concurrency in a shared file environment, multiple users should be permitted to read a file that is being concurrently updated. Also, one user should be permitted to update records in the same file in which another user is updating different records.

This excerpt simply recognizes that users should be permitted to update different records of a file at the same time. There is no disclosure or even suggestion that a process modifies at least one characteristic of a file without receiving the file.

At Col. 3; 8-22, Strickland states:

> In operation, the VSAM Record Level Sharing (RLS) methods of this invention permit a first process or Unit Of Work (UOW) to fetch a Private Buffer (PB) copy of a data CI containing the record of interest from shared Structured External Storage (SES) to a PB in Shared Local Cache (SLC). After the fetch, the first UOW logs all updates to the record of interest and conditionally writes the changed data CI back to SES. The SES write is conditioned upon there not having been an interim write to the same data CI by a second UOW. If a second UOW has indeed written the same data CI in the interim, the first UOW recycles by referring to SES for the most recent copy of the data CI, combining it with the logged updates from the first UOW and again conditionally writing the updated data CI back to SES.

Like the first excerpt, this excerpt discloses that a process should obtain its own private copy of a data CI before modifying it. Again, this is opposite to the claimed limitation which recites that the second process modifies the at least one characteristic of the file without receiving a copy of the file.

Applicant submits that Strickland, by describing a system in which a process obtains its own private copy of data before modifying it, actually teaches away from the claimed invention. Accordingly, Applicant submits that the combination of Harter and Strickland fails to render claim 1 obvious.

Independent claims 14, 19 and 30 likewise recite a first message that requests "a set" or "a plurality" of tokens, a second message granting "the set" or "the plurality" of tokens, and modifying at least one characteristic of a file without receiving a copy a copy of the file. Independent claim 27 recites a first message that requests "a set" or "a plurality" of tokens, a second message granting "the set" or "the plurality" of tokens. As set forth above, the combination of Harter and Strickland, which fail to disclose or even suggest these limitations, and which actually teach away from the claimed invention, does not render the claimed invention obvious. Accordingly, the rejection of claims 14, 19, 27 and 30 should also be reversed.

Claims 6, 20 and 33

Independent claim 6 recites in relevant part:

> "a first message that grants a set of tokens . . . the set of tokens being required for the second process to be able to modify at least one characteristic of a file", and
>
> "if the second process receives the set of tokens, the second process modifying the at least one characteristic of the file without receiving a copy of the file."

As set forth above, the combination of Harter and Strickland fails to render the invention recited in claim 6 obvious.

Independent claims 20 and 33 recite similar limitations. Therefore, the rejection of these claims should be reversed.

Claims 11, 21 and 36

Independent claim 11 recites in relevant part:

> "a first process . . . that generates a request to a second process for a grant **of a <u>set</u> of tokens** required to enable the first process to modify at least one characteristic of a file", and
>
> "if the first process receiving the set of tokens, the first process modifying the at least one characteristic of the file ... without receiving a copy of the file."

As set forth above in the analysis regarding claim 1, the combination of Harter and Strickland fails to render the invention recited in claim 11 obvious. Accordingly, the rejection of claim 11 should be reversed.

Independent claims 21 and 36 recite similar limitations. Thus, the rejection of these claims also should be reversed.

Claims 2-5, 7-10, 12, 13, 15-18, 22-26, 28, 29, 31, 32, 34, 35 and 37-40 depend from the above-mentioned independent claims. As the rejection of the independent claims should be reversed for the reasons set forth above, the rejection of these dependent claims should also be reversed.

In sum, Applicant submits that the final Office Action fails to establish a <u>prima</u> <u>facie</u> case of obviousness, and that therefore the rejection of claim 1-40 should be reversed.

CONCLUSION

Applicant respectfully submits that the claims are allowable over the art of record.

Accordingly, Applicant request that the rejection of all claims be reversed.

Respectfully submitted,

/Michael R. Reinemann/
Michael R. Reinemann
Reg. No. 38,280
Tel. 617-951-3060

Please direct all correspondence to:

IP Administration Legal Department,
M/S 35
Hewlett-Packard Co.
P.O. Box 272400
Fort Collins, CO 80527-2400

CLAIMS APPENDIX

(Claims on Appeal in Appl. Ser. No. 09/428,384)

1. A computerized data file system, comprising:

a first process that maintains a data file stored in a computer-readable memory; and

a second process that generates a first message requesting that said second process be granted by said first process a plurality of tokens required for said second process to modify at least one characteristic of said file stored in said computer-readable memory,

said first process generating a second message, in response to said first message, that grants said tokens to said second process if said tokens are available for grant to said second process, and

if said tokens are granted, said second process modifying the at least one characteristic of said data file as maintained by said first process in said computer-readable memory without said second process receiving a copy of said data file.

2. A system according to claim 1, wherein:

said first process is resident at a server computer node, and said second process is resident at a client computer node.

3. A system according to claim 1, wherein:

if any of said tokens are unavailable for grant to said second process as a result of current grant of said tokens to at least one other process, said first process generates a third message revoking the current grant of said tokens to said at least one other process.

4. A system according to claim 3, wherein:

said at least one other process, in response to said third message, generates a fourth message making said tokens available for grant by said first process.

5. A system according to claim 3, wherein:

said first process resides in a first computer node;

said second process resides in a second computer node;

said at least one other process resides in at least one other computer node; and

said first computer, second computer, and at least one other computer nodes are networked together and are remote from each other.

6. A computer node, comprising:

a first process residing in the node that generates a first message that grants a set of tokens, if the set of tokens is available for grant, to a second process that requested grant of the set of tokens, the set of tokens being required for the second process to be able to modify at least one characteristic of a file stored in a computer-readable memory within the computer node,

if the second process receives the set of tokens, the second process modifying the at least one characteristic of the file without receiving a copy of the file.

7. A node according to claim 6, wherein:

the second process resides in a remote computer node.

8. A node according to claim 7, wherein:

one of the first and second processes resides in a server computer node and the other of the processes resides in a client computer node.

9. A node according to claim 6, wherein:

if at least one token in the set of tokens is unavailable for grant because the at least one token is currently granted to a third process, the first process also generates a second message that revokes current grant of the at least one token to the third process prior to generating the first message.

10. A node according to claim 6, wherein:

the first message is generated by the first process in response to a request for the grant of the set of tokens generated by the second process, the request specifying all tokens required for the second process to be able to modify the at least one characteristic of the file.

11. A computer node, comprising:

a first process residing in said node that generates a request to a second process for grant of a set of tokens required to enable the first process to modify at least one characteristic of a file residing in a remote computer-readable memory,

if the first process receives the set of tokens, the first process modifying the at least one characteristic of the file residing in the remote computer-readable memory without receiving a copy of the file.

12. A node according to claim 11, wherein:

the second process resides in a second computer node, and the memory is comprised in said second node.

13. A node according to claim 11, wherein:

the set of tokens comprises all tokens required for the first process to be able to modify the at least one characteristic of the file.

14. A network computer system, comprising:

a first computer node having a data file stored in a computer-readable memory; and

a second computer node that issues to the first computer node a first message requesting grant of a set of tokens required to carry out a modification of at least one characteristic of the file stored in the first computer node,

the first computer node issuing a second message to the second computer node after receipt of the first message, the second message granting the set of tokens to the first process if the set of tokens is available for grant to the second process, and

if the set of tokens are granted, the second computer node modifying the at least one characteristic of the file stored in the first computer node without the second computer node receiving a copy of the file.

15. A system according to claim 14, wherein:

the first computer node is a server node, and the second computer node is a non-server node.

16. A system according to claim 14, wherein:

the set of tokens comprises all tokens required to carry out the modification of the at least one characteristic of the file.

17. A system according to claim 14, wherein:

if at least one token in the set of tokens is unavailable for the grant because the at least one token is currently granted, the first computer node waits to issue the first message until after the first computer node receives a third message from a third computer node indicating relinquishment of current grant of the at least one token.

18. A system according to claim 17, wherein:

the at least one token comprises a plurality of tokens.

19. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions maintaining a data file in a computer storage memory;

second instructions generating a first message requesting grant, to a process, of a plurality of tokens required to modify at least one characteristic of said file located in said computer storage memory;

third instructions generating a second message, in response to said first message, that grants said tokens if said tokens are available for grant to said process; and

fourth instructions, responsive to the plurality of tokens being granted, modifying the at least one characteristic of said data file located in said computer storage memory without said process receiving a copy of said data file.

20. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions generating a first message that grants a set of tokens, if the set of tokens is available for grant, to a requester of the set of tokens, the set of tokens being required to permit the requester to be able to modify at least one characteristic of a file stored in computer storage memory, and

second instructions, responsive to the set of tokens being granted to the requester, modifying the at least one characteristic of the file stored in the computer storage memory without the requester receiving a copy of said data file.

21. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions generating a request for grant of a set of tokens required to enable modification by an issuer of the request of at least one characteristic of a file residing in <u>a</u> storage memory; and second instructions, responsive to the set of tokens being granted to the issuer, modifying the at least one characteristic of the file residing in the storage memory without the issuer receiving a copy of the file.

22. Computer-readable memory according to Claim 19, further comprising:

further instructions generating a third message, if any of said tokens are unavailable for grant as a result of a current grant of said tokens, revoking the current grant of said tokens.

23. A computer-readable memory according to claim 22, wherein:

said further instructions, in response to said third message, generate a fourth message making said tokens available for grant.

24. Computer-readable memory according to claim 20, further comprising:

further instructions generating a second message, if at least one token in the set of tokens is unavailable for grant because the at least one token is currently granted, that revokes previous grant of the at least one token prior to generating the first message.

25. Computer-readable memory according to claim 20, wherein:

the first message is generated in response to a request for the grant of the set of tokens generated, the request specifying all tokens required to be able to modify the at least one characteristic of the file.

26. Computer-readable memory according to claim 21, wherein:

the set of tokens comprises all tokens required to be able to modify the at least one characteristic of the file.

27. A computerized data file system, comprising:

means for maintaining a data file stored in a computer-readable memory;

means for generating a first message requesting grant of a plurality of tokens required to modify at least one characteristic of said file stored in said computer-readable memory;

means for generating a second message, in response to said first message, that grants said tokens if said tokens are available for grant; and

means for modifying said data file stored in said computer-readable memory, if said tokens are granted.

28. A system according to claim 27, further comprising:

means for generating, if any of said tokens are unavailable for grant as a result of current grant of said tokens, a third message revoking the current grant of said tokens.

29. A system according to claim 28, further comprising:

means for generating, in response to said third message, a fourth message making said tokens available for grant.

30. A computerized method for coherently maintaining and modifying a data file, comprising:

maintaining said data file in a computer-readable memory;

generating a first message requesting grant of a plurality of tokens required to modify at least one characteristic of said data file in said computer-readable memory;

generating a second message, in response to said first message, that grants said tokens if said tokens are available for grant; and

if said tokens are granted, modifying said at least one characteristic of said data file in said computer-readable memory.

31. A method according to claim 30, further comprising:

if any of said tokens are unavailable for grant as a result of current grant of said tokens to at least one other process, generating a third message revoking the grant of said tokens.

32. A method according to claim 31, wherein:

in response to said third message, a fourth message making said tokens available for grant is generated.

33. A computerized method for use in maintaining coherency of a data file stored in a computer-readable memory, comprising:

generating a first message that grants a set of tokens, if the set of tokens is available for grant, to a requester of the grant of the set of tokens, the set of tokens being required for the requester to be able to modify at least one characteristic of the file stored in the computer-readable memory; and in response to the set of tokens being granted to the requester, modifying the at least one characteristic of the file stored in the computer-readable memory without the requester receiving a copy of the file.

34. A method according to claim 33, wherein:

if at least one token in the set of tokens is unavailable for grant because the at least one token has been currently granted, the method also comprises a second message that revokes current grant of the at least one token prior to generating the first message.

35. A method according to claim 33, wherein:

the first message is generated in response to a request for the grant of the set of tokens generated by the requester, the request specifying all tokens required for the requester to be able to modify the at least one characteristic of the file.

36. A computerized method for use in maintaining coherency of a data file stored in a computer-readable memory, comprising:

generating a request for grant of a set of tokens required to enable modification of at least one characteristic of the data file stored in the computer-readable memory; and

in response to the set of tokens being granted, modifying the at least one characteristic of the data file stored in the computer-readable memory.

37. A method according to claim 36, wherein:

the set of tokens comprises all tokens required to be able to modify the at least one characteristic of the file.

38. The system according to claim 1, wherein:

said second process, in response to receiving said second message, modifies said at least one characteristic of said file stored in said computer-readable memory.

39. The system according to claim 27, further comprising:

means for modifying said at least one characteristic of said file stored in said computer-readable memory.

40 The method according to claim 30, further comprising:

modifying said at least one characteristic of said file in said computer-readable memory.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.